## • Nombre de proyecto: *Inventarium universalis*

## • Objetivo del proyecto

Realizar un programa que facilite el registro de cantidades sobre diferentes nodos definidos por el usuario que permite agregar, editar y eliminar dichos nodos, así como guardarlos en un archivo para tener un registro de fácil edición y con un formato específico para que pueda ser leído por el programa, ahorrando tiempo al usuario.

## • Alcance de su proyecto

El proyecto cumple con la totalidad de las funciones en su alcance más básico, que es el manejo estructurado de nodos con cantidades predefinidas (aunque no exclusivas) y su guardado en archivos de manera local. Sin embargo, el proyecto es mucho más escalable si se usaran bases de datos directo a servidores para guardar una copia de los inventarios en la nube, facilitando así exportarlo a plataformas móviles y sincronizarlo con más personas, o usando tecnología NFC o códigos QR creando así mayores campos de aplicación tangible del algoritmo, cosas que debido a la limitación del curso no pueden ser aplicadas aún.

## • Introducción

El programa satisface la necesidad tener un control sobre qué y cuántos productos tiene a su disposición el usuario en el momento que sea requerido, además de dejar a nivel código la puerta abierta para modificaciones que se adapten mejor a las necesidades particulares de cada usuario. La importancia de desarrollar este proyecto es el tener un registro fiable, moderno y automático el cual pueda evitar errores que, dependiendo del uso, signifiquen exceso de productos, falta de los mismos, anomalías causadas por robos, que se traducen en pérdidas monetarias para el usuario, además de contar con un algoritmo de fácil aplicación en diversos sistemas, pudiendo incluso ser una de las funciones de un sistema aún mayor.

Las aplicaciones son muy amplias y variadas, ya que dependiendo de la escala a la que se piense utilizar, el programa se ajusta a los requerimientos del usuario. Va desde el sencillo inventario de las compras en el hogar, pasando por el inventario de un pequeño local, ya sea informal o formal que les ayude a llevar el registro de sus recursos, pasando por grandes cadenas como supermercados y fábricas, ya sea de alimentos, partes mecánicas, entre una infinidad. Todo aquello que se requiera contar y obtener una lista de manera estructurada, puede ser cubierto con este programa.

## • Desarrollo

El proyecto fue realizado a partir de plantearse la solución a una pequeña problemática doméstica que se mantenía constante: al momento de realizar las compras, siempre se olvidaba algún producto debido a que no se tenía un registro eficiente de los productos con los que contábamos en casa. Esto generaba discusiones innecesarias y quería evitarlas con ayuda de un sencillo programa. Sin embargo, conforme fue avanzando el desarrollo del proyecto, se observó que la solución que se estaba implementando era demasiado grande para la problemática original, por lo que se amplió su objetivo y ahora satisfacía problemáticas a nivel empresarial, pues pueden ingresarse diversas características, como lo fue el precio, y dependiendo del usuario, pueden agregarse fácilmente “n” características más para guardar y mostrar (de ahí su nombre en latín, “inventario universal”, pues cual palabra, puedes ir a las raíces (código) y adaptarse a cualquier tipo de inventario que se desee).

Se utilizó el lenguaje de programación C, ya que debido a la formación que tengo es el que conozco a más profundidad, además de que debido a que es un lenguaje de programación de nivel medio, el proyecto aprovecha las ventajas de la programación de alto y bajo nivel, pues facilita la programación además de aprovechar el manejo directo de la memoria por medio de apuntadores, necesarios para implementar las estructuras de datos. El proyecto fue realizado utilizando la estructura utilizada fue una lista ligada, ya que debido a sus características satisface de manera óptima las necesidades del programa, pues le da la característica dinámica que las colas o pilas no permiten, pues no tiene un tamaño fijo, y se utilizó para crear cada miembro ingresado por el usuario, facilitando así su guardado, impresión y manejo de la información. Además, se optimizó el uso de memoria para únicamente abrir y guardar los archivos tras realizar todas las modificaciones que el usuario necesite, evitando también posibles errores humanos.

### o Algoritmo

1. Crear un nodo con una cadena alfanumérica llamada item, un número entero llamado cantidad y un número real llamado precio
2. Repetir mientras el usuario desee
   1. Crear una lista.
   2. r = ‘1’
   3. Mientras r =‘1’
      1. Mostrar “Seleccione una opción: Crear nuevo inventario ó Cargar inventario ó Salir ó Ayuda”
      2. Solicitar un carácter y almacenarlo en OpcionArchivo.
      3. Si OpcionArchivo = “Crear nuevo inventario”
         1. Mostrar “Se creará un nuevo archivo”
         2. r = ‘0’
      4. Si OpcionArchivo = “Cargar inventario”
         1. Solicitar una cadena alfanumérica y almacenarla en NombreArchivo
         2. Si NombreArchivo existe
            1. Repetir hasta final del archivo

INSERTAR (lista, item, cantidad, precio)

* + - 1. En caso contrario mostrar “No existe el archivo.”
      2. r = ‘0’
    1. Si OpcionArchivo = “Salir”
       1. Opcion = “Salir”
       2. r = ‘0’
    2. Si OpcionArchivo = “Ayuda”
       1. Mostrar AYUDA
    3. En caso contrario mostrar “Opción no válida”
  1. Repetir mientras Opcion sea diferente de “Salir”
     1. Mostrar “Elija: Insertar | Buscar | Modificar | Eliminar | Mostrar | Cerrar | Guardar | Ayuda”
     2. Solicitar un carácter y almacenarlo en Opcion
     3. Si Opcion = ‘Insertar’
        1. Mientras desee repetirse
           1. Solicitar una cadena, un entero y un real y almacenarlo en item, cantidad y precio
           2. INSERTAR (lista, item, cantidad, precio)
           3. Preguntar si desea repetir
     4. Si Opcion = ‘Buscar’
        1. Mientras desee repetirse
           1. Solicitar una cadena alfanumérica y almacenarla en ítemBuscado
           2. BUSCAR (lista,item,itemBuscado)
           3. Si la lista no está vacía

Sí item = itemBuscado entonces mostrar item, cantidad, precio

En caso contrario mostrar “Nodo no encontrado”

* + - * 1. En caso contrario mostrar “Lista vacía”
        2. Preguntar si desea repetir
    1. Si Opcion = ‘Modificar’
       1. Mientras desee repetirse
          1. Solicitar una cadena alfanumérica y almacenarla en ítemBuscado
          2. BUSCAR (lista,item,itemBuscado)
          3. Si la lista no está vacía

Sí item = itemBuscado

Mostrar “Elemento a modificar: A-Nombre B-Cantidad C-Precio”

Solicitar un carácter y almacenarlo en OpcionM

Si OpcionM = ‘A’ entonces leer cadena y almacenarla en item

Si OpcionM = ‘B’ entonces leer entero y almacenarlo en cantidad

Si OpcionM = ‘C’ entonces leer real y almacenarlo en precio

En caso contrario mostrar “Opción no válida”

En caso contrario mostrar “Nodo no encontrado”

* + - * 1. En caso contrario mostrar “Lista vacía”
        2. Preguntar si desea repetir
    1. Si Opcion = ‘Eliminar’
       1. Mientras desee repetirse
          1. Solicitar una cadena alfanumérica y almacenarla en ítemBuscado
          2. BUSCAR (lista,item,itemBuscado)
          3. Si la lista no está vacía entonces ELIMINAR (lista, nodo)
          4. En caso contrario mostrar “Lista vacía”
          5. Preguntar si desea repetir
    2. Si Opcion = ‘Mostrar’
       1. Si la lista no está vacía
          1. Repetir mientras HEAD no sea NULL

Mostrar producto, cantidad, precio

HEAD = NEXT

* + - 1. En caso contrario mostrar “Lista vacía”
    1. Si Opcion = ‘Guardar’
       1. Si OpcionArchivo = ’Cargar inventario’
          1. Mostrar “Si desea guardar el inventario con el mismo nombre, presione 1”
          2. Solicitar un carácter y almacenarlo en Sobreescribir
       2. Si Sobreescribir es diferente de ‘1’ entonces solicitar una cadena alfanumérica y almacenarla en NombreArchivo
       3. Si la lista no está vacía
          1. Repetir mientras HEAD no sea NULL

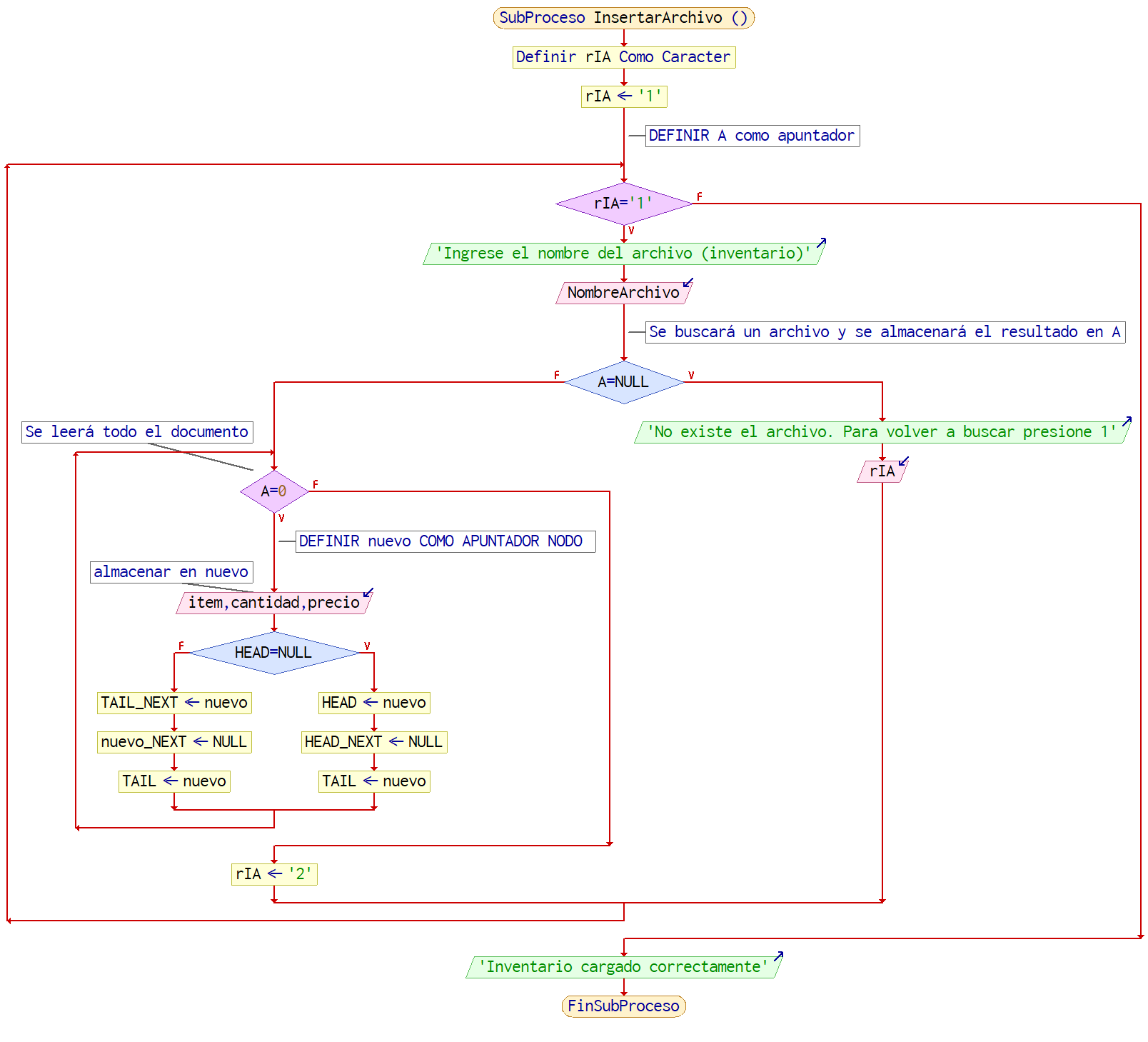
Escribir producto, cantidad, precio

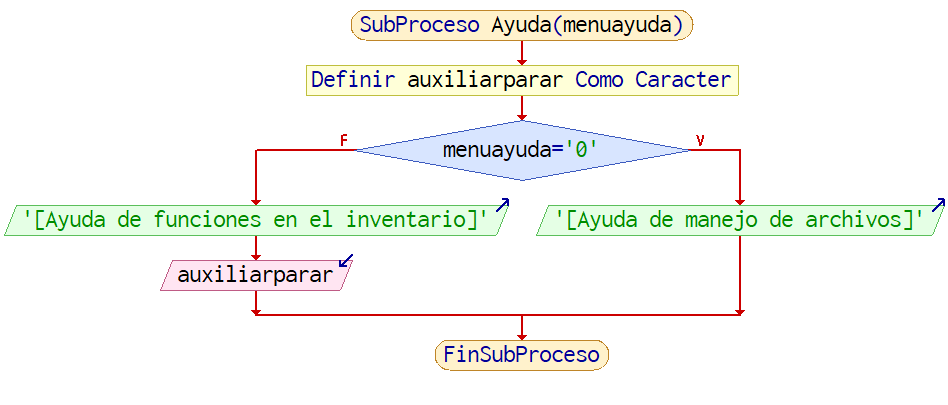
HEAD = NEXT

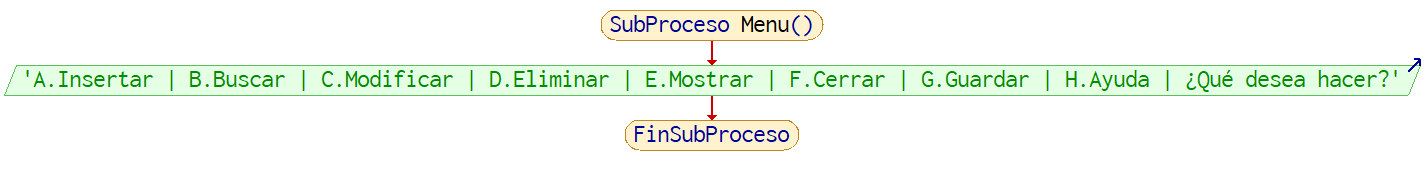
* + - 1. En caso contrario mostrar “Lista vacía”
    1. Si Opcion = ‘Ayuda’ entonces mostrar AYUDA
    2. En caso contrario mostrar “Opción no válida”
  1. Opcion = ‘0’
  2. Preguntar si desea repetir el proceso
  3. Si Iniciar = ’Verdadero’ entonces vaciar lista

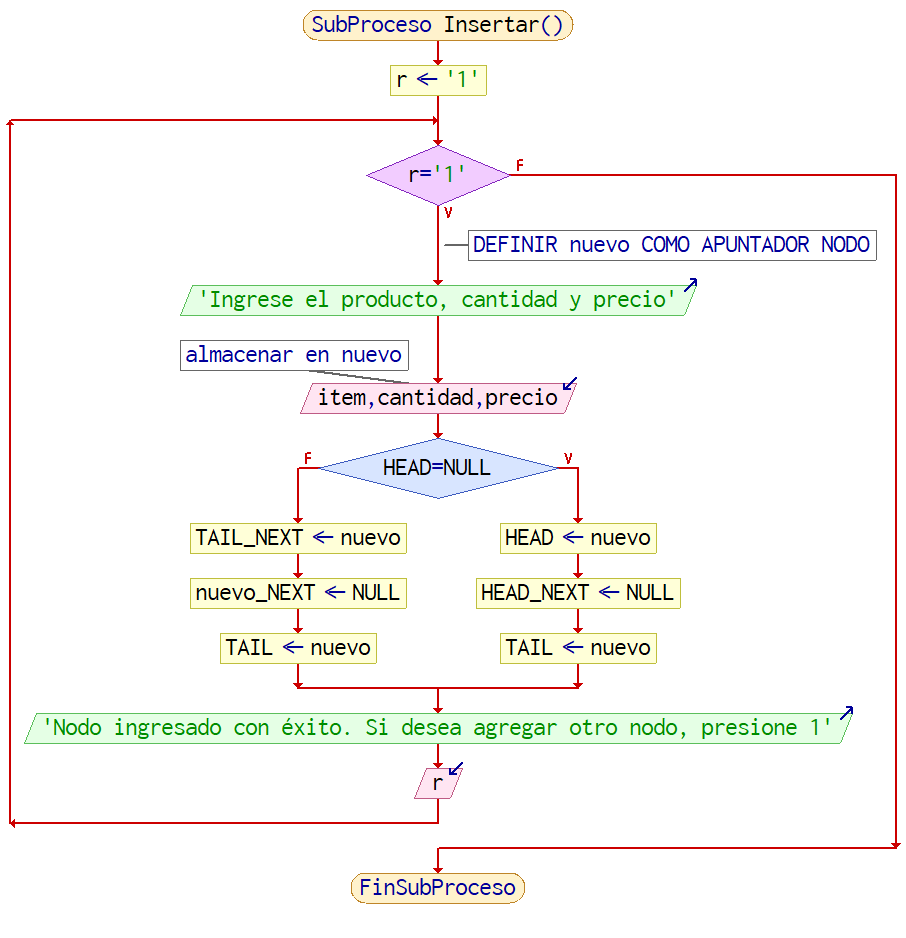
### o Diagrama de Flujo

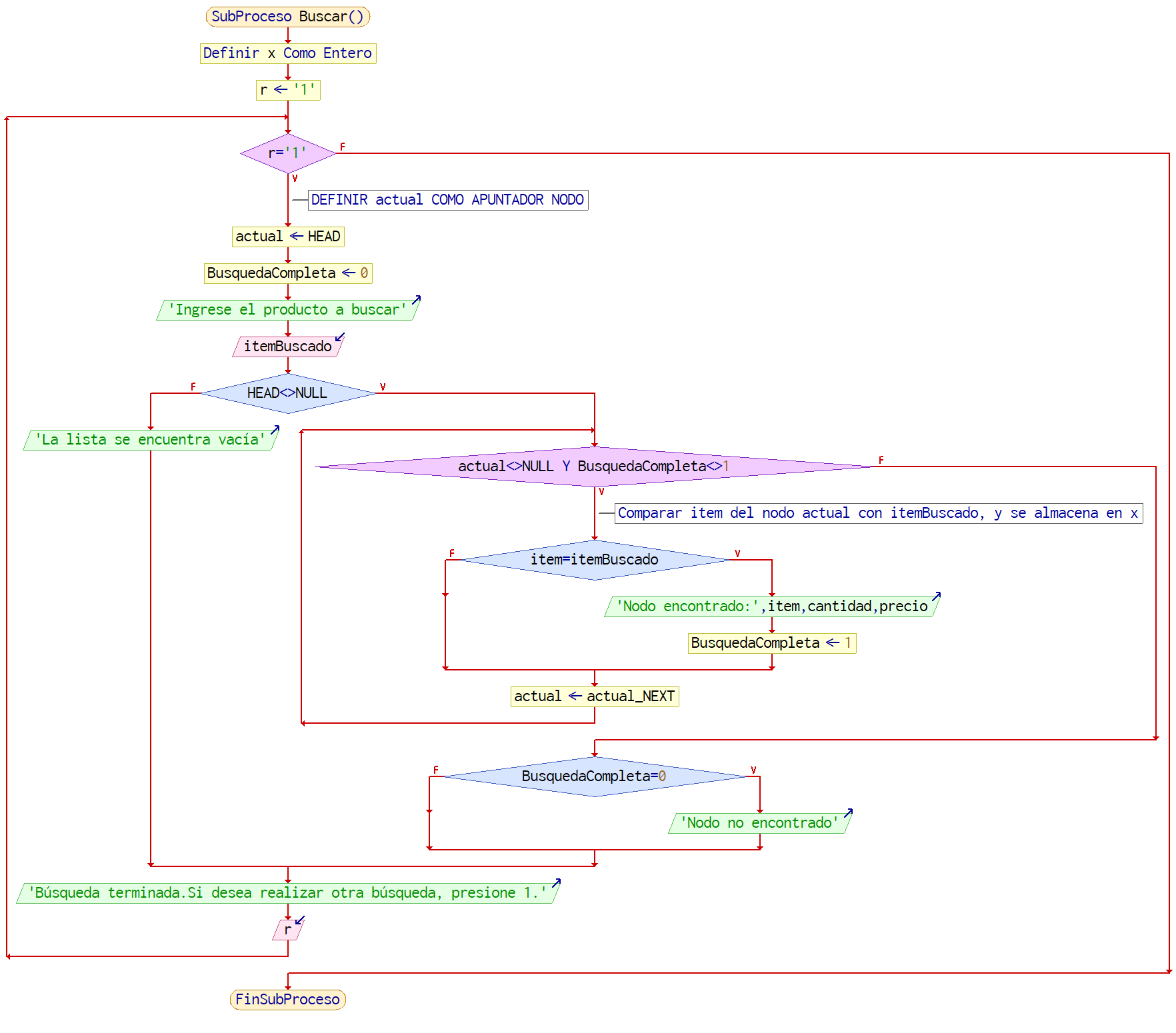


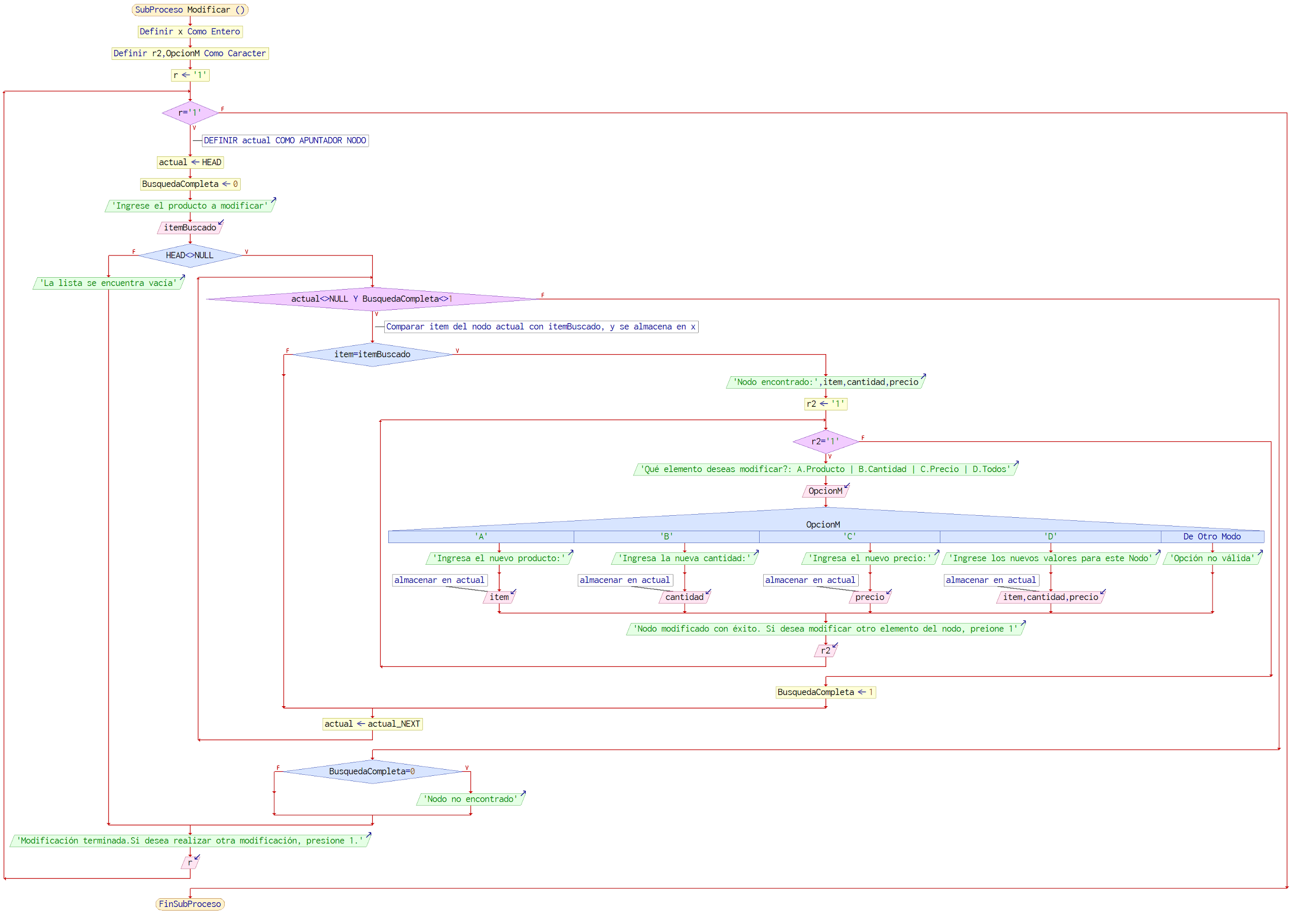


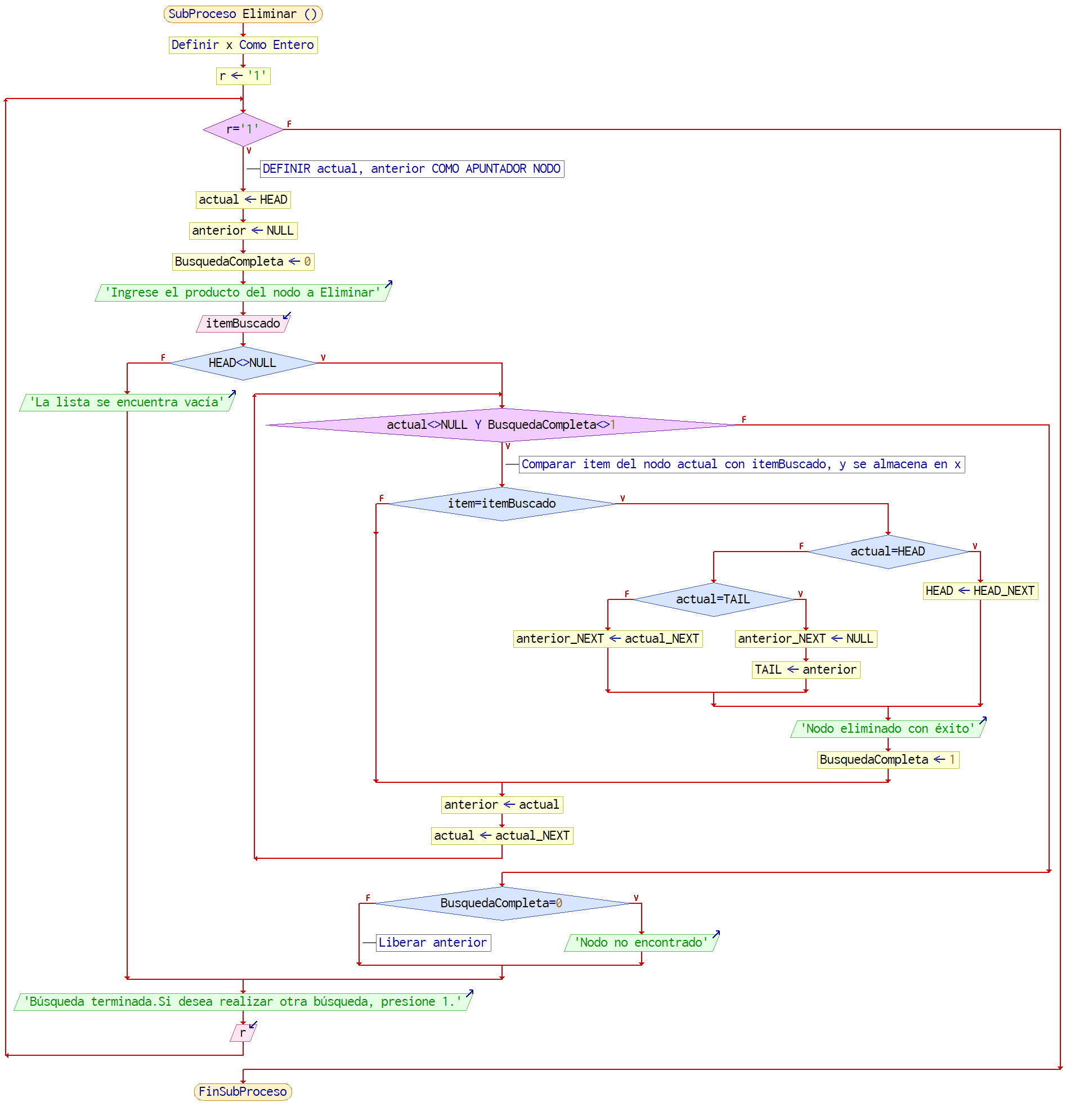


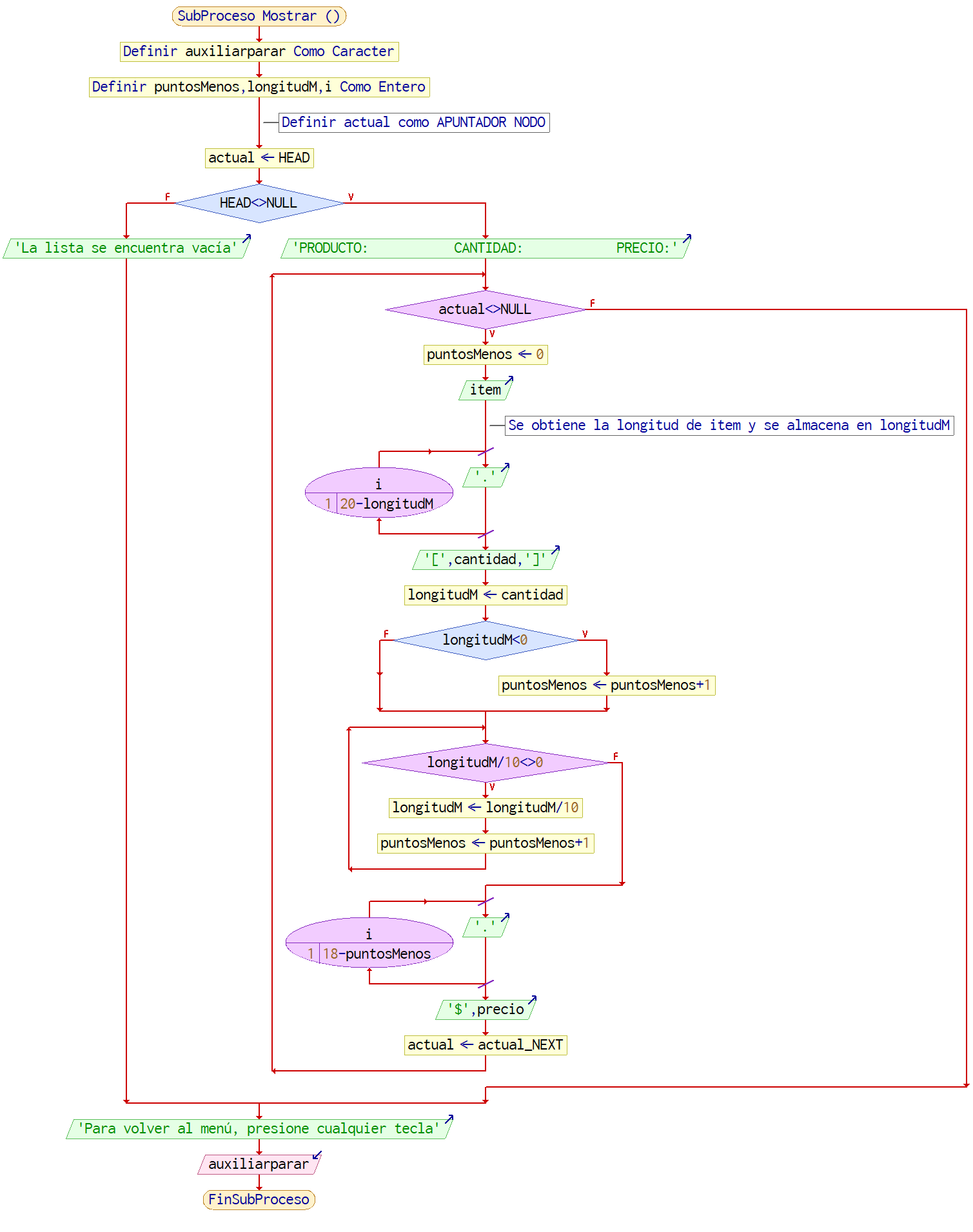


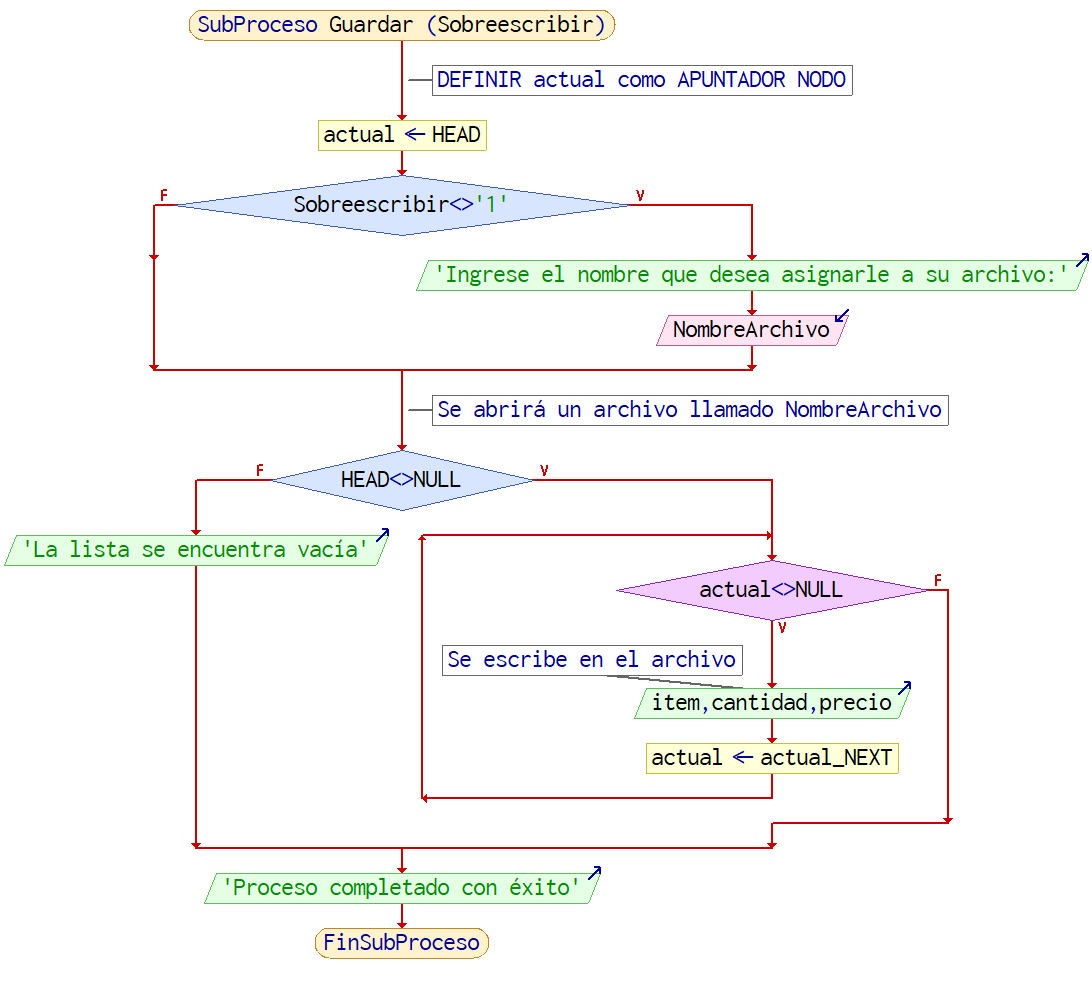












Los diagramas pueden ser vistos a detalle en: <https://github.com/RodrigoPG-W/Proyecto_Final/tree/master/Proyecto%20Final%20EDAI/Pseudoc%C3%B3digo%20y%20Diagrama%20PSeInt>

### o Pseudocódigo

**INICIO** **PROCESO Principal**

DEFINIR OpcionSobreescribir,Iniciar,r,OpcionArchivo,Opcion como CARÁCTER

DEFINIR cantidad,xc,yc,BusquedaCompleta como ENTERO

DEFINIR precio como REAL

DEFINIR item,NombreArchivo,itemBuscado como CADENAS

DEFINIR HEAD,TAIL,HEAD\_NEXT,TAIL\_NEXT,actual,anterior,actual\_NEXT,anterior\_NEXT, NULL como APUNTADOR NODO

Iniciar := '1'

MIENTRAS Iniciar = '1' HACER

HEAD := NULL

TAIL := NULL

r := '1'

ESCRIBIR 'Bienvenid@!'

MIENTRAS r = '1' HACER

ESCRIBIR 'Seleccione una opción: A-Crear nuevo inventario | B-Cargar inventario | C-Salir | D-Ayuda'

LEER OpcionArchivo

SEGÚN OpcionArchivo HACER

'A':

ESCRIBIR 'Se creará un nuevo archivo'

r := '0'

'B':

**InsertarArchivo()**

r := '0'

'C':

Opcion := 'F'

r := '0'

'D':

Opcion := '0'

**Ayuda(Opcion)**

DE OTRO MODO:

ESCRIBIR 'Opción no válida'

FIN SEGÚN

FIN MIENTRAS

MIENTRAS Opcion <> 'F' HACER

**Menu()**

LEER Opcion

SEGÚN Opcion HACER

'A':

**Insertar()**

'B':

**Buscar()**

'C':

**Modificar()**

'D':

**Eliminar()**

'E':

**Escribir ()**

'F':

'G':

SI OpcionArchivo = 'B' ENTONCES

ESCRIBIR 'Si desea guardar con el mismo nombre, presione 1'

LEER OpcionSobreescribir

FIN SI

**Guardar(OpcionSobreescribir)**

'H':

**Ayuda(Opcion)**

Opcion := '0'

DE OTRO MODO:

ESCRIBIR 'Opción no válida'

FIN SEGÚN

FIN MIENTRAS

Opcion := '0'

ESCRIBIR 'Gracias por usar el programa. Si desea abrir otro inventario, presione 1'

LEER Iniciar

FIN MIENTRAS

**FIN PROCESO**

**SUBPROCESO InsertarArchivo ()**

DEFINIR rIA como CARÁCTER

rIA := '1'

DEFINIR A como APUNTADOR

MIENTRAS rIA = '1' HACER

ESCRIBIR 'Ingrese el nombre del archivo (inventario)'

LEER NombreArchivo

//Se buscará un archivo y se almacenará el resultado en A

SI A = NULL ENTONCES

ESCRIBIR 'No existe el archivo. Para volver a buscar presione 1'

LEER rIA

SÍ NO

MIENTRAS A=0 HACER // Se leerá todo el documento

DEFINIR nuevo como APUNTADOR NODO

LEER item,cantidad,precio // almacenar en nuevo

SI HEAD = NULL ENTONCES

HEAD := nuevo

HEAD\_NEXT := NULL

TAIL := nuevo

SÍ NO

TAIL\_NEXT := nuevo

nuevo\_NEXT := NULL

TAIL := nuevo

FIN SI

FIN MIENTRAS

rIA := '2'

FIN SI

FIN MIENTRAS

ESCRIBIR 'Inventario cargado correctamente'

**FIN SUBPROCESO**

**SUBPROCESO Ayuda(menuayuda)**

DEFINIR auxiliarparar como CARÁCTER

SI menuayuda = '0' ENTONCES

ESCRIBIR '[Ayuda de manejo de archivos]'

SÍ NO

ESCRIBIR '[Ayuda de funciones en el inventario]'

LEER auxiliarparar

FIN SI

**FIN SUBPROCESO**

**SUBPROCESO Menu()**

ESCRIBIR 'A.Insertar | B.Buscar | C.Modificar | D.Eliminar | E.Mostrar | F.Cerrar | G.Guardar | H.Ayuda | ¿Qué desea hacer?'

**FIN SUBPROCESO**

**SUBPROCESO Insertar()**

r := '1'

MIENTRAS r='1' HACER

DEFINIR nuevo como APUNTADOR NODO

ESCRIBIR 'Ingrese el producto, cantidad y precio'

LEER item,cantidad,precio // almacenar en nuevo

SI HEAD = NULL ENTONCES

HEAD := nuevo

HEAD\_NEXT := NULL

TAIL := nuevo

SÍ NO

TAIL\_NEXT := nuevo

nuevo\_NEXT := NULL

TAIL := nuevo

FIN SI

ESCRIBIR 'Nodo ingresado con éxito. Si desea agregar otro nodo, presione 1'

LEER r

FIN MIENTRAS

**FIN SUBPROCESO**

**SUBPROCESO Buscar()**

DEFINIR x como ENTERO

r := '1'

MIENTRAS r = '1' HACER

DEFINIR actual como APUNTADOR NODO

actual := HEAD

BusquedaCompleta := 0

ESCRIBIR 'Ingrese el producto a buscar'

LEER itemBuscado

SI HEAD <> NULL ENTONCES

MIENTRAS actual <> NULL Y BusquedaCompleta <> 1 HACER

// Comparar item del nodo actual con itemBuscado, y se almacena en x

SI item = itemBuscado ENTONCES

ESCRIBIR 'Nodo encontrado:',item,cantidad,precio

BusquedaCompleta := 1

FIN SI

actual := actual\_NEXT

FIN MIENTRAS

SI BusquedaCompleta = 0 ENTONCES

ESCRIBIR 'Nodo no encontrado'

FIN SI

SÍ NO

ESCRIBIR 'La lista se encuentra vacía'

FIN SI

ESCRIBIR 'Búsqueda terminada.Si desea realizar otra búsqueda, presione 1.'

LEER r

FIN MIENTRAS

**FIN SUBPROCESO**

**SUBPROCESO Modificar ()**

DEFINIR x como ENTERO

DEFINIR r2,OpcionM como CARÁCTER

r := '1'

MIENTRAS r = '1' HACER

DEFINIR actual como APUNTADOR NODO

actual := HEAD

BusquedaCompleta := 0

ESCRIBIR 'Ingrese el producto a modificar'

LEER itemBuscado

SI HEAD <> NULL ENTONCES

MIENTRAS actual <> NULL Y BusquedaCompleta <> 1 HACER

// Comparar item del nodo actual con itemBuscado, y se almacena en x.

SI item = itemBuscado ENTONCES

ESCRIBIR 'Nodo encontrado:',item,cantidad,precio

r2 := '1'

MIENTRAS r2 = '1' HACER

ESCRIBIR 'Qué elemento deseas modificar?: A.Producto | B.Cantidad | C.Precio | D.Todos'

LEER OpcionM

SEGÚN OpcionM HACER

'A':

ESCRIBIR 'Ingresa el nuevo producto:'

LEER item // almacenar en actual

'B':

ESCRIBIR 'Ingresa la nueva cantidad:'

LEER cantidad // almacenar en actual

'C':

ESCRIBIR 'Ingresa el nuevo precio:'

LEER precio // almacenar en actual

'D':

ESCRIBIR 'Ingrese los nuevos valores para este Nodo'

LEER item,cantidad,precio // almacenar en actual

DE OTRO MODO:

ESCRIBIR 'Opción no válida'

FIN SEGÚN

ESCRIBIR 'Nodo modificado con éxito. Si desea modificar otro elemento del nodo, presione 1'

LEER r2

FIN MIENTRAS

BusquedaCompleta := 1

FIN SI

actual := actual\_NEXT

FIN MIENTRAS

SI BusquedaCompleta = 0 ENTONCES

ESCRIBIR 'Nodo no encontrado'

FIN SI

SÍ NO

ESCRIBIR 'La lista se encuentra vacía'

FIN SI

ESCRIBIR 'Modificación terminada.Si desea realizar otra modificación, presione 1.'

LEER r

FIN MIENTRAS

**FIN SUBPROCESO**

**SUBPROCESO Eliminar ()**

DEFINIR x como ENTERO

r := '1'

MIENTRAS r = '1' HACER

DEFINIR actual, anterior como APUNTADOR NODO

actual := HEAD

anterior := NULL

BusquedaCompleta := 0

ESCRIBIR 'Ingrese el producto del nodo a Eliminar'

LEER itemBuscado

SI HEAD <> NULL ENTONCES

MIENTRAS actual <> NULL Y BusquedaCompleta <> 1 HACER

// Comparar item del nodo actual con itemBuscado, y se almacena en x

SI item = itemBuscado ENTONCES

SI actual = HEAD ENTONCES

HEAD := HEAD\_NEXT

SÍ NO

SI actual = TAIL ENTONCES

anterior\_NEXT := NULL

TAIL := anterior

SÍ NO

anterior\_NEXT := actual\_NEXT

FIN SI

FIN SI

ESCRIBIR 'Nodo eliminado con éxito'

BusquedaCompleta := 1

FIN SI

anterior := actual

actual := actual\_NEXT

FIN MIENTRAS

SI BusquedaCompleta = 0 ENTONCES

ESCRIBIR 'Nodo no encontrado'

SÍ NO

// Liberar anterior

FIN SI

SÍ NO

ESCRIBIR 'La lista se encuentra vacía'

FIN SI

ESCRIBIR 'Búsqueda terminada.Si desea realizar otra búsqueda, presione 1.'

LEER r

FIN MIENTRAS

**FIN SUBPROCESO**

**SUBPROCESO Mostrar ()**

DEFINIR auxiliarparar como CARÁCTER

DEFINIR puntosMenos,longitudM,i como ENTERO

DEFINIR actual como APUNTADOR NODO

actual := HEAD

SI HEAD <> NULL ENTONCES

ESCRIBIR 'PRODUCTO: CANTIDAD: PRECIO:'

MIENTRAS actual <> NULL HACER

puntosMenos := 0

ESCRIBIR item

// Se obtiene la longitud de item y se almacena en longitudM

PARA i := 1 HASTA 20-longitudM HACER

ESCRIBIR '.'

FIN PARA

ESCRIBIR '[',cantidad,']'

longitudM := cantidad

SI longitudM < 0 ENTONCES

puntosMenos := puntosMenos+1

FIN SI

MIENTRAS longitudM/10 <> 0 HACER

longitudM := longitudM/10

puntosMenos := puntosMenos+1

FIN MIENTRAS

PARA i := 1 HASTA 18-puntosMenos HACER

ESCRIBIR '.'

FIN PARA

ESCRIBIR '$',precio

actual := actual\_NEXT

FIN MIENTRAS

SÍ NO

ESCRIBIR 'La lista se encuentra vacía'

FIN SI

ESCRIBIR 'Para volver al menú, presione cualquier tecla'

LEER auxiliarparar

**FIN SUBPROCESO**

**SUBPROCESO Guardar (Sobreescribir)**

DEFINIR actual como APUNTADOR NODO

actual := HEAD

SI Sobreescribir <> '1' ENTONCES

ESCRIBIR 'Ingrese el nombre que desea asignarle a su archivo:'

LEER NombreArchivo

FIN SI

// Se abrirá un archivo llamado NombreArchivo

SI HEAD <> NULL ENTONCES

MIENTRAS actual <> NULL HACER

ESCRIBIR item,cantidad,precio // Se escribe en el archivo

actual := actual\_NEXT

FIN MIENTRAS

SÍ NO

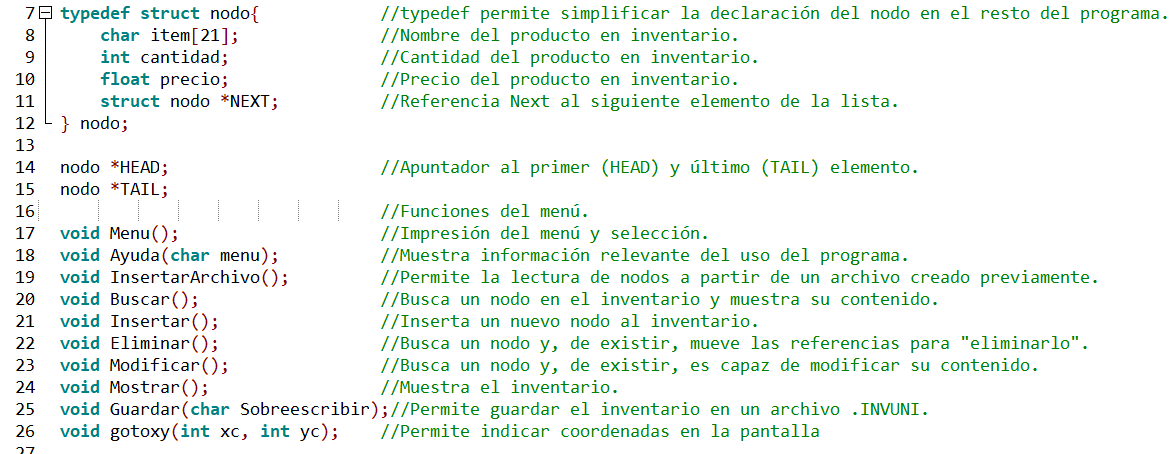
ESCRIBIR 'La lista se encuentra vacía'

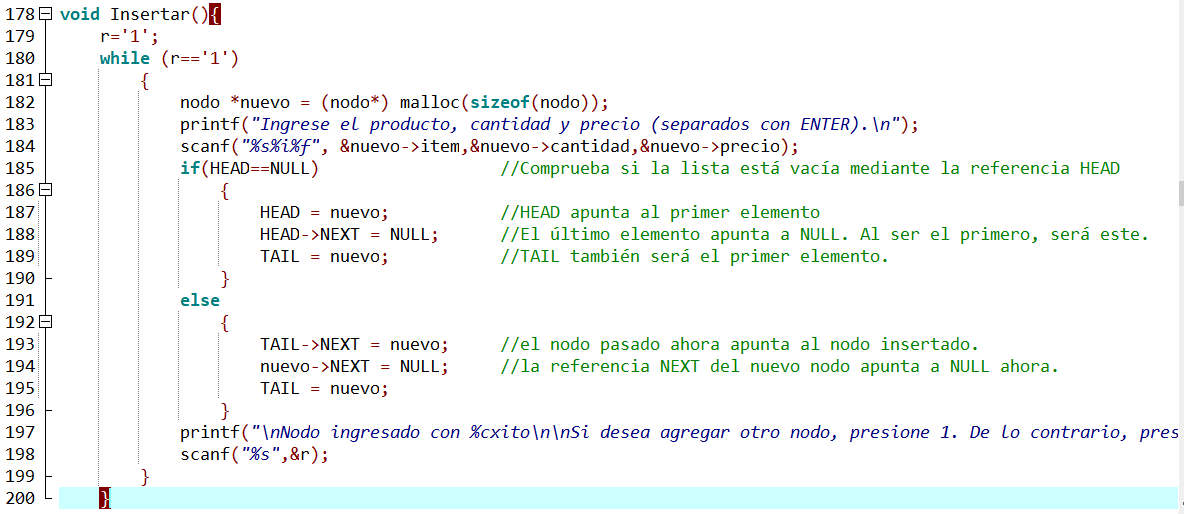
FIN SI

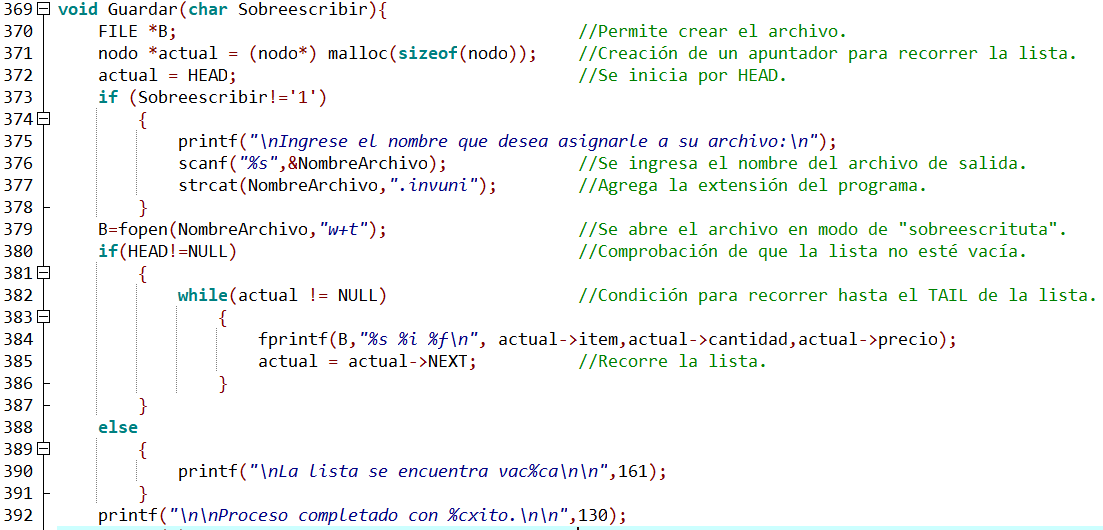
ESCRIBIR 'Proceso completado con éxito'

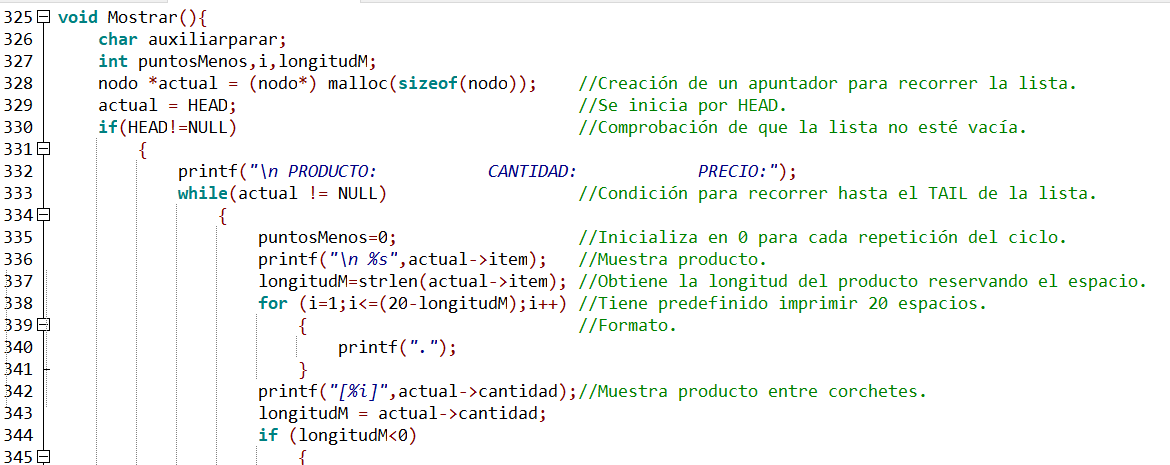
**FIN SUBPROCESO**

### o Código (sólo algunos pantallazos)

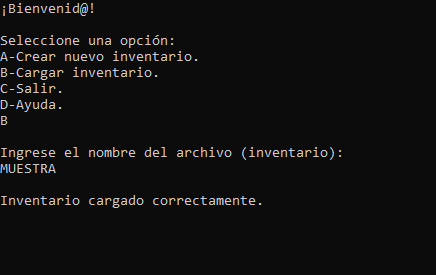


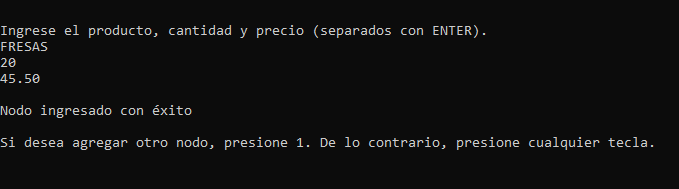


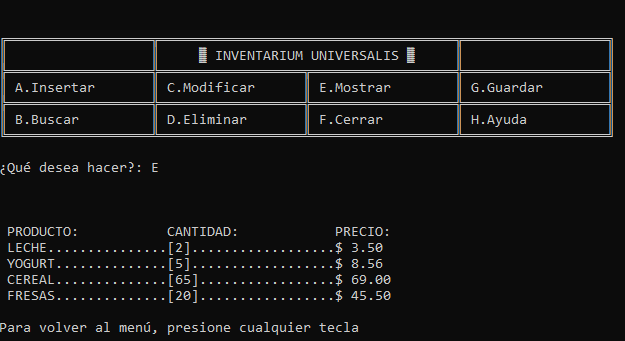


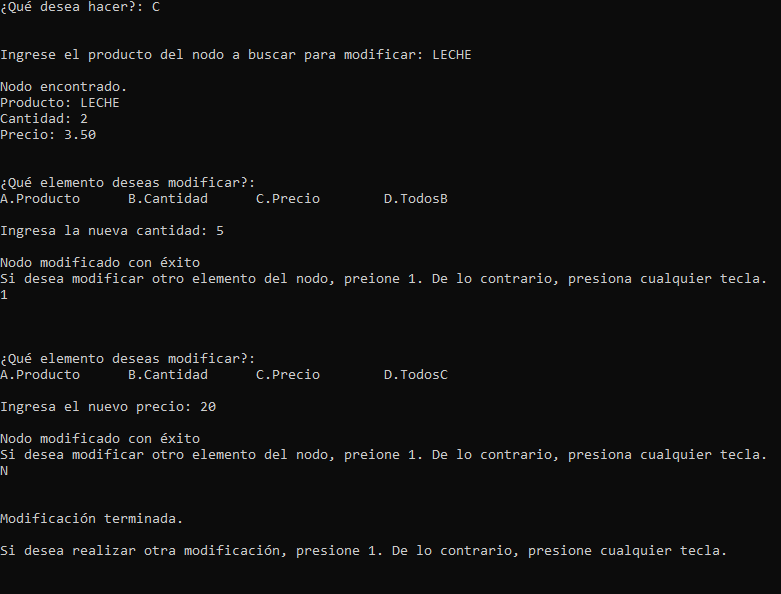


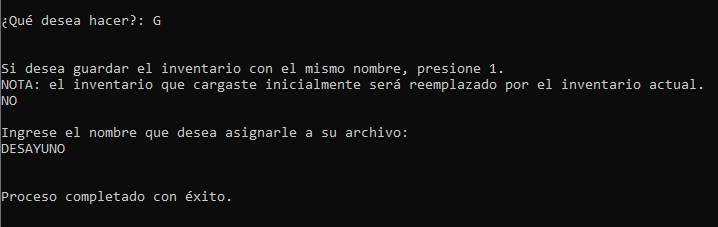
## • Resultados

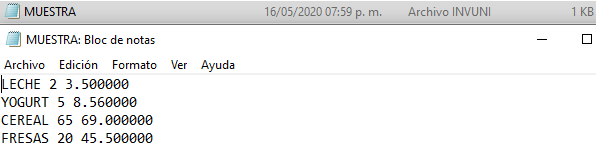












## • Presentación:

<https://www.youtube.com/watch?v=-E9IQU-VDrY>

A partir de la demostración del software: <https://youtu.be/-E9IQU-VDrY?t=373>

## • Conclusiones

En la programación, uno de los retos que se afronta es el buscar la solución óptima a una infinidad de problemas, teniendo como herramienta únicamente los conocimientos del programador, pudiendo dar así una infinidad de soluciones a un problema; sin embargo, no todas son necesariamente la mejor ruta a ejecutar. Con lo aprendido durante el curso, pude analizar algunas metodologías útiles tanto para la abstracción de un problema como también para dar una mejor solución al mismo, utilizando un lenguaje tan versátil como lo es C y aprovechar así el manejo de memoria en el equipo. Gracias a ello, pude desarrollar un pequeño software capaz de mantener de manera ordenada un registro de cualquier cosa que se desee inventariar, solucionando así una problemática más que puede escalarse tan alto como se pueda concebir.

Nunca había desarrollado un proyecto tan grande por mi cuenta, aunque tras finalizar (de momento) su desarrollo y tras un poco de reflexión, este proyecto me dejó con nuevas ideas sobre la programación, la más importante a mi parecer es el hecho de que el software que programé no es tan grandioso por sí mismo si hablamos de él únicamente como un programa en lenguaje C. Tiene aún muchas limitaciones si lo comparamos con programas similares como lo es Excel o incluso llevar un registro manual, sin embargo, la ventaja que tiene y que sin duda voy a explotar más a fondo conforme mis conocimientos se amplían es su “algoritmo”. El valor de la lógica detrás de él es mucho más importante que la codificación en un lenguaje.

Si bien es cierto que también es necesaria (pues de nada sirve tener una solución si no se puede ejecutar) el hecho de saber exactamente cómo funciona la metodología, como se debe de comportar, cada línea y el porqué está estructurado de la forma en la que está es lo que lo hace valioso y versátil, pues podría enumerar un sinfín de modificaciones y en dónde hacerlas para hacerlo cada vez mejor y adaptarlo para cada cliente que pudiese utilizar mi software, el cual puede dar la ventaja de disminuir errores humanos y evitar el tener que aprender complicadas operaciones en otros programas, por lo que considero que el programa fue un éxito en mi formación como programador, y cuyo desarrollo únicamente queda pendiente por un tiempo, pues estoy ansioso de implementar todas las ideas que surgieron en la parte del diseño y desafortunadamente tuvieron que ser reservadas por diversos factores.

## • Referencias

* Jorge A. Solano (2019), “Manual de prácticas del laboratorio de Estructuras de datos y algoritmos I” (consultado el 03/04/2020). Recuperado de: <http://lcp02.fi-b.unam.mx/>
* Andrés Vergara (2016), “Estructura de datos en C” (consultado el 03/04/2020). Recuperado de: <https://www.youtube.com/channel/UCnjiTpRymNvwcsW0yQ1JY4Q>
* Jesús Lucas (2019), “*Qué es C: Características y sintaxis*” (consultado el 01/05/2020). Recuperado de: <https://openwebinars.net/blog/que-es-c/>
* Theon Weber (2017), “*Las diferencias entre los lenguajes de programación de alto y de bajo nivel*” (consultado el 01/05/2020). Recuperado de: <https://techlandia.com/tipos-lenguaje-computadoras-lista_104385/>
* Quezada, Cintia (2019), “Apuntes de Fundamentos de Programación”. Facultad de Ingeniería, México.
* Revista Énfasis Logística (2017),” Urgen a empresas actualizar operaciones en almacenes” (consultado el 24/05/2020). Recuperdo de: <http://www.logisticamx.enfasis.com/notas/77623-urgen-empresas-actualizar-operaciones-almacenes>